Dagstuhl Seminar 01141, 2–6 April, 2001
Semantic Foundations of Proof-search

David Pym Eike Ritter Thomas Streicher

May 2, 2001

1 Introduction

We present a brief report on Dagstuhl Seminar 01141, "Semantic Foundations of Proof-search", held 2–6 April, 2001. We begin, in § 2, with the usual description of the seminar, as required by the Dagstuhl administration. Then, in § 3, we provide a brief account of the seminar itself, discussing the main themes emerging from the lectures and the associated conversations. In § 4 we give the programme of the seminar. In § 5, we give the abstracts of the lectures delivered by the particpants. We conclude with a list of the participants and their addresses.

2 Description

Traditionally, logics are formulated as systems of deductive inference in which proofs are constructions which derive conclusions from given assumptions. However, in computing, many problems are naturally formulated as questions of reductive inference in which the correctness of a given putative conclusion must be shown by reduction, commonly formulated as proof-search in a given formal system, to established acceptable assumptions. Examples of this phenomenon include type-inference, parsing, program correctness and internet information retrieval. Typically, such examples are described as long and complex formal texts. Consequently, algorithmic proof-search is a fundamental enabling technology throughout the computing sciences. Moreover, the reductive view of inference represents an alternative view of logic, just as fundamental as the deductive one, which is largely undeveloped.

So far, the theory of proof-search has developed mostly along proof-theoretic lines but using many type-theoretic techniques. The utility of type-theoretic methods suggests that semantic methods of the kind found to be valuable in the semantics of programming languages should be useful in tackling the main outstanding difficulty in the theory of proof-search, i.e.,

the representation of intermediate stages in the search for a proof. The space of searches is much larger than the space of proofs: An adequate semantics would represent both the space of searches and the space of proofs and give an account of the recovery of proofs (which are extensional objects) from searches (which are more intensional objects). It would distinguish between different proof-search strategies and permit analyses of their relative merits.

This seminar helps to establish a program to build such a semantics. To this end, we propose the following foci for the seminar:

- Reductive vs. deductive logic: their logical, mathematical and computational properties;
- Proof-search in type-theoretic languages: the role of typing constraints during proof-search;
- Proof- and model-theoretic analyses of search spaces: the search-oriented counterparts to traditional proof theory and model theory;
- Intensional semantics for proof-search: specific intensional and computational models based on structures such as games, continuations and realizability;
- Applications of proof-theoretic and semantic techniques to the design and implementation of theorem provers.

3 The Seminar

The seminar was lively and friendly, with many people commenting that they found the exposure to some new ideas quite stimulating. It was particularly pleasing that there was little or no tendency among the participants to form into subgroups: Everyone talked to everyone else.

Several broad themes may be identified in the given lectures:

- Foundational issues: Basic questions about the meaning and mathematical semantics of search spaces and search-objects;
- Logic programming: Issues in semantics and pragmatics;
- Type theory and interactive theorem proving: Issues in the formulation and representation of problems;
- Tableaux and counter-models:
- Syntactic methods: Optimizing the execution dynamics of search engines via the logical properties of the target system;
- Applications: To formal mathematics, to logic programming and to verification of Java bytecode!

4 Programme of the Seminar

Monday, 2 April

David Pym	*Semantic foundations of proof-search*
Conor McBride	*Elimination with a motive*
Kurt Ranalter	*A decision procedure and Kripke-style semantics for the language ILP of intuitionistic formal pragmatics*
Andrea Schalk	*Games played on graphs*
Dominique Larchey-Wendling	*Refutation as counter-models for intuitionistic logic*
Roy Dyckhoff	*Rule invertibility in sequent calculi*

Tuesday, 3 April

Eike Ritter	*Modelling backtracking*
Didier Galmiche	*Proof-search and counter-model generation in mixed logics*
Edmund Robinson	*Proof-nets for classical logic*
Alberto Momigliano	*Uniform proof-search and negation*
Uwe Egly	*A polynomial translation of propositional S4 into propositional intuitionistic logic*
Gianluigi Bellin	*Towards a pragmatics*

Wednesday, 4 April

Andrei Voronkov	*Syntactic foundations of proof-search*
Iliano Cervesato	*A linear spine calculus*
James McKinna	*Towards a calculus of problems in type theory with applications to relational rippling*

Thursday, 5 April

Jim Lipton	*Semantics of higher-order logic programming*
Patricia Hill	*Logic programming, data-flow analysis and equality theories*
Robert Stärk	*Problems of bytecode verification*
Thomas Streicher	*Ontological status of paraproofs*
Kevin Watkins	*Defining an operational semantics for resource management in Lolli*
Alan Smaill	*Research specification language in practice*
Randy Pollack	*On the extensibility of proof-checkers*
Konstantin Korovin	*Solving Knuth-Bendix ordering constraints*

Friday, 6 April

Manfred Kerber	*Towards a classification of proofs*
Alexandre Riazanov	*Splitting without backtracking*
Roy Dyckhoff	*Proof-search and computation in Herbelin's calculus*

5 Abstracts

Semantic foundations of proof-search

David Pym
Queen Mary, University of London

Algorithmic proof-search is an essential enabling technology throughout informatics. Proof-search is the proof-theoretic realization of the formulation of logic not as a theory of deduction but rather as a theory of reduction. Whilst deductive logics typically have a well-developed semantics of proofs, reductive logics are typically well-understood only operationally. Each deductive system can, typically, be read as a corresponding reductive system. We discuss some of the problems which must be addressed in order to provide such a semantics of proof-searches of comparable value to the corresponding semantics of proofs. Just as the semantics of proofs is intimately related to the model theory of the underlying logic, so too should be the semantics of proof-searches. We discuss how it is that the semantics of search is essentially constructive, in the sense of Kripke, and how to solve the problem of providing a semantics for (the construction of) proof-searches which adequately models both the operational and logical aspects of the reductive logic. In particular, we propose an algebraic approach to the semantics of control.

(*cf.* ENTCS 37(2000), 18pp.)

Elimination with a motive

Conor McBride
University of Durham

Elimination rules for datatypes look like this:

$$\forall \Phi(\forall \vec{y} \cdots \to \Phi \vec{t}[\vec{y}]) \to \cdots \forall \vec{x}.\Phi \vec{x}$$

In a refinement proof we exploit our ability to choose Φ to attack"the goal at hand". Φ abstracts the motive for the elimination. The subgoals so generated not only expose the "predecessors" of the $\vec{x}$, they also yield more specific applications of Φ, rewriting the goal in the light of the new information available. Such rules explain the leverage that the assumptions listed in the $\vec{x}$ give us on an arbitrary goal.

I give a tactic to apply rules in this style, and I use this tactic not only with the native rules supplied with datatypes and relations, but also with derived rules of the above form. The tactic takes any such theorem (suitably annotated) and turns it into a left rule in the sense of the sequent calculus.

Examples of such rules are used to

- give an analysis of induction which generates case analysis from recursion
- characterize the behaviour of functions
- equip datatypes with a more sophisticated notion of pattern and smaller subsystem

A decision procedure and Kripke-style semantics for the language ILP of Intuitionistic Formal Pragmatics

Kurt Ranalter
University of Verona

We consider the language ILP of Intuitionistic Formal Pragmatics proposed by Bellin and Dalla Pozza (Dagstuhl Seminar on Linear Logic, 1999) focusing on the relation between the notions of causal implication, assertability and obligation. We give a Kripke-style semantics ILP and prove the completeness theorem for it. The decision procedure extends standard procedures for intuitionistic logic with one for a fragment of relevant implicational logic and of deontic logic: it takes a sequent S as input and yields either a proof of S in the sequent calculus for ILP or a countermodel for S.

Games played on graphs

Andrea Schalk
University of Manchester

Games have become increasingly successful when it comes to modelling a variety of logics and programming languages. These games are typically played on trees - once play has diverged, two strands can never be reunited. This does not agree wit ha description of games via positions, where it is of no importance how a given constellation was arrived at. Furthermore, even for the simples games (eg Hyland's "Games for Fun", see CLICS Summer School Lecture Notes, Cambridge 1995) the categorical structure is not as simple as one might like - the linear function space of two such simple tree games can become quite complex. Choosing positions as a primitive concept we introduce games played on graphs rather than trees. We keep the standard restrictions, that is there are two players, P and O, who move strictly alternatingly, and O always starts. We show that we can define a linear function space - its positions are elements of the product of the set of positions of the constituent games - allowing us to define a category in the usual way with morphisms being given by strategies on the linear function space of A and B. This category is symmetric monoidal closed and has all products. A full subcatgegory has the property that morphisms can be seen

as relations, and composition (originally defined via "parallel composition and hiding") becomes relational composition. We further define a "Curien exponential" giving rise to a cartesian closed category in the standard way. This exponential formalizes the idea that O should be allowed to explore a given strategy (for P) on A and satisfies the property that it produces a finite games when applied to a finite game, unlike the standard construction for tree games. We conclude in identifying applications of this model.

Refutation as counter-models for intuitionistic logic

Dominique Larchey-Wendling
LORIA, Nancy

We propose to investigate and to reuse the semantics of intuitionistic linear logic (ILL), from an initial analysis of known semantics like phase semantics or Petri net semantics. Thus, we focus on notions like quantale, closure and resource frames, and we define a new semantics of ILL that is called resource semantics. The completeness and finite model property are proved from a based-on proof search method in which countermodels are obtained from refutation trees. Moreover, we define a new preordered monoid semantics from an adequate choice of pretopology. As Petri Nets can be seen as a concrete representation of preordered monoids, such a choice also leads to a new Petri nets semantics for ILL with new results like completeness and finite model property. From these semantical considerations we obtain some results about non-provability in ILL and then we can expect to develop methods for the generation of countermodels.

This is joint work with Didier Galmiche.

Rule invertibility in sequent calculi

Roy Dyckhoff
University of St Andrews

We discuss a simple calculus for Gödel-Dummett logic in which all inference rules are invertible, illustrating the point that rule invertibility (for the avoidance of backtracking) is related not to the use of classical logic but to the use of a logic characterized by linear (i.e., non-branching) Kripke models.

Modelling backtracking

Eike Ritter
University of Birmingham

The work presented in this talk is part of a bigger project, which intends to give semantics to proof search. In this talk we present some general steps

which are necessary to achieve such a semantics, and focus on one aspect, namely how to model backtracking in intuitionistic logic via continuations.

The first step consists of giving semantics to partial, possibly incompleteable proofs. We use polynomial categories for this purpose. The universal property of these categories ensures that a partial proof can be completed to a proof if and only if one can find a substitution consisting only of ground terms for the indeterminates in the polynomial category. The left-rules of the sequent calculus force us to consider also a Kripke-style semantics where the information contained in the Kripke-worlds is the substitution arising from modelling the implication left-rule.

In this paper we model backtracking by embedding intuitionistic logic into classical logic. Hence in a second step we extend these polynomial categories and the Kripke-semantics to the $\lambda\mu\nu$-calculus, a term calculus for classical logic. In this semantics, a switch in the focus on the right-hand side corresponds to applying a continuation in functional languages.

When we embed LJ-proofs with backtracking into LK and translate these proofs into the semantics we have developed, we realize that backtracking in LJ-proofs gives rise to switching the right-hand side in the LK-proofs resulting from the embedding, and hence to continuations.

This is joint work with David Pym.

Proof-search and counter-model generation in mixed logics

Didier Galmiche
LORIA, Nancy

We study proof-search in the propositional BI-logic that can be viewed as a merging of intuitionistic logic and multiplicative intuitionistic linear logic with its underlying sharing interpretation. BI is the basis of new foundations for Computer Science applications (logic programming, reasoning about mutable data structures). We propose a labelled tableau calculus for BI, the use of labels making it possible to generate countermodels. We show that from a given formula A, a non-redundant tableau construction procedure terminates and yields either a tableau proof of A or a finite countermodel of A in terms of the Kripke resource monoid semantics. Moreover, we prove the finite model property for BI with respect to this semantics.

Proof nets for classical logic

Edmund Robinson
Queen Mary, University of London

Classical sequent calculus is a beautifully symmetric system but imposes an ordering on operations which makes too many distinctions from a semantic point of view. In this talk I propose to outline how someone seeking a more

graphical representation is led to something like proof nets, and to explain why the only serious problem in adapting the standard linear technology is caused by weakening. Taking weakening as a rule in its own right, not as a nullary contraction, allows an extension of the standard Danos-Regnier switchings to cover it. This gives a notion of proof net for full classical propositional logic. We sketch the proof that any such net is sequentializable (i.e. can be derived from a valid sequent proof). In this talk I shall only deal with statics.

A "negative" look to uniform proof-search

Alberto Momigliano
University of Leicester

Logical frameworks with a logic programming interpretation, such as hereditary Harrop formulae (HHF), do not allow to express directly negative information, in order to preserve goal-oriented proof search. Since negation-as-failure does not fit well in a logical framework especially one endowed with hypothetical and parametric judgments, we adapt the idea of elimination of negation in Horn logic to a fragment of higher-order HHF. This entails finding a middle ground between the Closed World Assumption usually associated with negation and the Open World Assumption typical of logical frameworks; the main technical idea is to isolate a set of programs where static and dynamic clauses do not overlap.

A Polynomial Translation of Propositional S4 into Propositional Intuitionistic Logic

Uwe Egly
Technical University Vienna

We present a polynomial translation of the propositional fragment of the modal logic S4 into the propositional fragment of intuitionistic logic. The translation is performed in three main steps. Properties of intermediate translations are established by purely proof-theoretical means, i.e., by proof transformations between different cut-free sequent calculi. Consequently, this approach yields effective translation procedures.

Towards a formal pragmatics

Gianluigi Bellin
University of Verona

The project of providing a logic for formal pragmatics (presented in the Dagstuhl seminar on linear logic 1999) is motivated by the need of extending

the framework of logical theory to account for aspects of informal reasoning involving the illocutionary acts of assertion, obligation, etc. also in mixed contexts. A long-term aim of the project is to give "intended interpretations" of substructual logics and a coherent account of these formal system within the standard framework of logical theory. Currently, the language ILP has the usual intuitionistic connectives plus a connective -o expressing relevant causal implication; the elementary sentences are of the form $\vdash \alpha$ (α is assertible) and $O - \alpha$ (α is obligatory), where α is a classically interpreted proposition. The main result reached so far is the characterization of mixed-contexts inferences yielding the principle: "if α is obligatory and from the assertion that α it can be causally inferred the assertion that β, then β is obligator". An open problem is how to reflect classical reasoning within the system, e.g., to account for the principle "if β is forbidden and from the assertion that α it can be causally inferred the assertion that β, then α is forbidden".

Syntactic foundations of proof-search

Andrei Voronkov
University of Manchester

We show how to decide modal logics by the inverse method. Unlike the semantics-based tableau method, the inverse method only derives tautologies, so search space pruning using semantics is difficult, if possible at all. We demonstrate how one can find redundancy criteria for sequents and derivations using syntactic properties of derivations in sequent calculi.

A linear spine calculus

Iliano Cervesato
ITT Inc.

We present the spine calculus $S^{\rightarrow -\circ \& T}$ as an efficient representation for the linear lambda-calculus $\lambda^{\rightarrow -\circ \& T}$ which includes intuitionistic functions ($\rightarrow$), linear functions ($-\circ$), additive pairing ($\&$), and additive unit (T). enhances the representation of Church's simply typed lambda-calculus as abstract Böhm trees by enforcing extensionality and by incorporating linear constructs. This approach permits procedures such as unification to retain the efficient head access that characterizes first-order term languages without the overhead of performing eta-conversions at run time. Potential applications lie in proof search, logic programming, and logical frameworks based on linear type theories. We define the spine calculus, give translations of $\lambda^{\rightarrow -\circ \& T}$ into $S^{\rightarrow -\circ \& T}$ and vice-versa, prove their soundness and completeness with respect to typing and reductions, and show that the spine calculus is strongly normalizing and admits unique canonical forms.

Towards a calculus of problems in type theory with applications to relational rippling

James McKinna
University of Durham

We seek a (search) calculus of problems posed in a type theory with inductive definition. In particular, one which

- is distinguished from existing presentations of type theory
- emphasises induction/recursion/case analysis over search oriented analyses which focus on the lambda-Pi fragment of dependent type theory

We identify a general form of problem statement which subsumes the ϵ-expressions of Bundy and Lombart's Relational Rippling (Dagstuhl 9350; July 1995). We further identify a number of rewrite rules in relational rippling as arising from inversion of inductive definitions.

Induction, recursion, inversion, and even derived case analysis as in the work of McBride, are expressed as left rules in our calculus, whose general from is that of rewriting of subcontexts, similarly to the annotated rewriting of Bundy/Lombart.

Relational rippling in general is a notationally and semantically complex theory of rewriting annotated ϵ-expressions. Our hope is, by reanalysing these notions in our new framework, we may simplify as well as learn from, this important search technique. This is very much work in progress.

Semantics of higher-order logic programming

Jim Lipton
Wesleyan University

Church's Theory is one of the older type theories (1940) based on (Church's) simply typed lambda calculus, with logical constants added at the appropriate types. It is the basis for lambda-prolog (Miller,Nadathur, Pfenning, Scedrov), as well as Peter Andrews' work on higher-order automated deduction in the 60s and 70s. Its classical semantics was developed in Henkin's thesis in 1950, and an intensional variant by Andrews in the 1970s. Using an indexed version of (Kripke-like) Omega-set models, we develop sound and complete semantics for the intuitionistic fragment of Church's theory of types (ICTT). We then show how to adapt the semantics to the lambda-prolog fragment(HOHH)of this theory via Uniform Algebras, a Goal-Atom-Program sorted variant of the above semantics. The problem of producing sound and complete semantics for Higher Order Hereditarily Harrop formulas with resolution (uniform) proofs had been open for ten years. We have solved it, but have not touched the question of lambda-prolog's rather

unique brand of polymorphism or its kind and type definition mechanisms. The mathematical background required is quite basic (a little intuitionistic semantics, an affection for completeness theorems).

The work was done jointly with Mary de Marco at Wesleyan.

Logic programming, data-flow analysis and equality theories

Patricia Hill
University of Leeds

For logic programming, the standard procedural semantics is based on resolution and, more specifically, *full unification*, which is sound with respect to the Herbrand equality theory. On the other hand, most Prolog systems implement a form of unification that is sound with respect to the equality theory of rational trees. The advantages are that the unification is more efficient and that the equality theory allows for cyclic terms which have many applications, including, in particular, an efficient representation of grammars. However, to really enjoy these benefits, we need to be able to support a flexible system that allows for both the Herbrand and the rational tree equality theory to exist side-by-side in the program. In particular, the system should help the programmer and compiler ensure that the unification at each step is sound for the intended equality theories and that uses of the data-structures are consistent with their cardinality and structure. In this note we discuss what is needed in more detail and show how data-flow analysers can provide this support.

Problems of bytecode verification

Robert Stärk
ETH Zürich

Most research on Java Bytecode Verification is focussed on the soundness of the bytecode verifier. From the security point of view it is important to know that bytecode which is accepted by the verifier does not violate type conditions at run-time. From the practical point of view it is also important to know that bytecode that is generated by a Java compiler from legal well-typed Java programs is accepted by the verifier afterwards. During an attempt to prove that a Java compiler generates verifiable code, however, we found examples of legal Java programs which are rejected by any Bytecode Verifier. The examples show that Java Bytecode Verification as it has been introduced by Sun is not possible. We propose therefore to restrict the so-called rules of definite assignment for the try-finally statement as well as for the labeled statement such that our example programs are no longer

allowed. Then we can prove, using the framework of Abstract State Machines, that each program from the restricted Java language is verifiable by the Java Bytecode Verifier.

Ontological status of paraproofs

Thomas Streicher
University of Darmstadt

Both in proof-search and proof theory certain paraproofs show up, i.e., derivations where some leaves are not justified by axioms but by authority. These justifications by authority rather serve the purpose of error elements known from programming. These paraproofs do not serve in general the purpose of establishing truths but rather are used for testing the real proofs.

This is basic to Girard's Ludics programme. But paraproofs also appear in recent work of Krivine and Danos on realizability for classical AF_2 and in Curien and Herbelin's Duality of Computation where paraproofs appear as continuation terms. We leave it as a question for future investigations to find out whether paraproofs may provide a bridge between the fields of proof search and continuation semantics.

Defining An Operational Semantics for Resource Management in Lolli

Kevin Watkins
Carnegie-Mellon University

A research specification language in practice

Alan Smaill
University of Edinburgh

We have been working for some time on a search engine that allows declarations of the primitive search steps and supports use of different search regimes by means of a small control language. One level of this is intended to give a general search mechanism in terms of a calculus of structural goals and means of achieving search goals using higher-order representations. I speculate on the integration of search and logic declarations within a single framework logic, and how semantic considerations should help this enterprise.

On the extensibility of proof-checkers

Randy Pollack
University of Durham

In a proof checker, we are interested in proofs; no other evidence for a judgement is acceptable. There is a question, however, whether we insist on objects that are immediately recognizable as proofs, or will accept some meta-notations that only compute to proofs. For example the deduction theorem is a sound rule (e.g. admissible) of Hilbert style minimal implicational logic, because a proof of $G \vdash a \to b$ can be computed from a proof of $G, a \vdash b$, but it is not directly a rule of the logic. Even if we allow a proof checker to directly implement such rules, we cannot expect to know and implement all the sound rules users may want. This talk is about how to construct a proof checker such that the stringent requirement of formality can be met without sacrificing the convenience and computational efficiency of such indirect proof notations as admissible rules.

Isn't this just what LCF tactics are for? Surprisingly, although LCF tactics are programmed in a Turing-complete programming language (e.g. SML), only derivable rules are representable. Further, because SML's type system is not expressive enough to specify which theorem a tactic will return, and because SML has general recursion which may not terminate, tactics must actually be evaluated, which is infeasible for many rules.

My proposal is to program LCF-style proof checkers in languages with only terminating functions, and dependent type systems. Then many non-derivable rules can be expressed as tactics, and these tactics for admissible rules can be used without being executed, because they cannot fail to terminate if executed, and their type tells explicitly which theorem they will return.

Solving Knuth-Bendix ordering constraints

Konstantin Korovin
University of Manchester

Solving ordering constrains is used in automated deduction to prune the search space. Mainly two kinds of orders are used; the Knuth-Bendix orders and recursive path orders. For recursive path orders, decidability of constraint solving was shown by Comon, and NP-completeness by Nieuwenhuis. We show that for the Knuth-Bendix order the constraint solvability problem is NP-complete. For constraints consisting of a single inequality we present a polynomial time algorithm. More information can be found at http://www.cs.man.ac.uk/ korovink.

Towards a classification of proofs

Manfred Kerber
University of Birmingham

In this talk different views on the concept of proof are presented. I advocate that the traditional view of Hilbert is only partly describing a multicoloured concept. Recent advances in the mechanization of proof (in particular, the integration of computer algebra systems and mechanized reasoning systems) makes it necessary to deal with very long proofs. In agent-oriented proof search it is also necessary to communicate partial proofs efficiently. In order to do that effectively and efficiently it is necessary to write and communicate proofs on different levels. In the presentation I discuss important different dimensions of proofs, in particular

- the formal system in which the proof is formulated (like natural deduction, tableau, resolution);
- the rigour of the proof (in Polya's terminology to distinguish between demonstrative proofs and plausible proofs);
- the problem formulation (is the theorem proved in the system it is originally stated or in a reformulated way);
- the completeness of the proof (whether it is given in full detail or whether a lot of computation is required to construct it);
- whether it is concise (the most concise form of a proof can be a triple consisting of the theorem, a theorem prover, and the number of steps performed by the prover to construct a proof);
- the probabilistic level (primality tests, e.g., often demonstrate primality with high probability, but do not guarantee a property);
- the information content (some faulty proofs can be easily patched, others do not contain any useful information).

Splitting without backtracking

Alexandre Riazanov
University of Manchester

The case analysis principle is the core of tableau-based theorem proving. Is can also be integrated into saturation-based theorem proving in the form of the splitting rule $S \bigcup \{C \vee D\} \rightarrow S \bigcup \{C\} \cup S \bigcup \{D\}$ if C and D have no common variables (in order to refute $S \bigcup \{C \vee D\}$, refute separately $S \bigcup \{C\}$ and $S \bigcup \{D\}$). Splitting can be implemented by using backtracking but this is difficult and affects the design of the system. We introduce a much simpler form of splitting that does not use backtracking: $S \bigcup \{C \vee D\} \rightarrow S \bigcup \{C \vee p, \neg P \vee D\}$ where p is a new predicate. This form of splitting is implemented in our resolution-and superposition-based theorem prover Vampire. Some optimizations to this technique are presented and experimentally compared.

For more information see A. Riazanov, A. Voronkov, *Splitting without Backtracking*, University of Manchester, CSPP-10.

Proof-search and computation in Herbelin's calculus

Roy Dyckhoff
University of St Andrews

Herbelin presented at CSL'94 a simple sequent calculus for minimal implicational logic, with a complete set of cut-reduction rules which is both confluent and strongly normalizing. We describe briefly its utility for proof search and its relationship to calculi for uniform proof search; our novel point is that if one adds cut-reduction rules to allow simulation of ordinary beta-reduction one has a system which is still confluent and strongly normalizing, and in fact preserves strong normalizability of untyped terms, thus solving a problem left open by Herbelin. We speculate about the possibility of extending the calculus to allow confluence on open terms ("meta-confluence") but retaining the SN-property.

This is joint work with Christian Urban, Cambridge.

6 The Participants

Gianluigi Bellin	University of Verona
Iliano Cervesato	ITT Inc. - Alexandria
Ewen Denney	University of Edinburgh
Roy Dyckhoff	University of St Andrews
Uwe Egly	TU Wien
Didier Galmiche	LORIA - Nancy
Patricia Hill	University of Leeds
Manfred Kerber	University of Birmingham
Konstantin Korovin	Manchester University
Kurt Ranalter	University of Verona
Dominique Larchey-Wendling	LORIA - Nancy
Jim Lipton	Wesleyan Univ. - Middletown
Conor McBride	University of Durham
James McKinna	University of Durham
Daniel Mery	LORIA - Nancy
Alberto Momigliano	University of Leicester
Michel Parigot	Université Paris VII
Randy Pollack	University of Durham
David Pym	Queen Mary College - London
Alexandre Riazanov	Manchester University
Eike Ritter	University of Birmingham
Edmund Robinson	Queen Mary College - London
Andrea Schalk	Manchester University
Alan Smaill	University of Edinburgh
Robert Stärk	ETH Zürich
Thomas Streicher	TU Darmstadt
Hayo Thielecke	University of Birmingham
Andrei Voronkov	Manchester University
Kevin Watkins	CMU - Pittsburgh